1.INTRODUCTION

1.1 OVERVIEW

Lung cancer is the leading cause of cancer-related deaths in the world. In the United States, lung cancer claims more lives every year than colon cancer, prostate cancer, and breast cancer combined.

The American Cancer Society's estimates for lung cancer in the United States for 2018 are:

- About 234,030 new cases of lung cancer (121,680 in men and 112,350 in women)
- About 154,050 deaths from lung cancer (83,550 in men and 70,500 in women)

1.2 PURPOSE

Despite the very serious prognosis (outlook) of lung cancer, some people with earlier-stage cancers are cured. More than 430,000 people alive today have been diagnosed with lung cancer at some point. The data is dedicated to classification problems related to the post-operative life expectancy in lung cancer patients: class 1 - death within one year after surgery, class 2 - survival.

We will be using classification algorithms such as Decision tree, Random Forest, KNN, and xgboost. We will train and test the data with these algorithms. From this best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

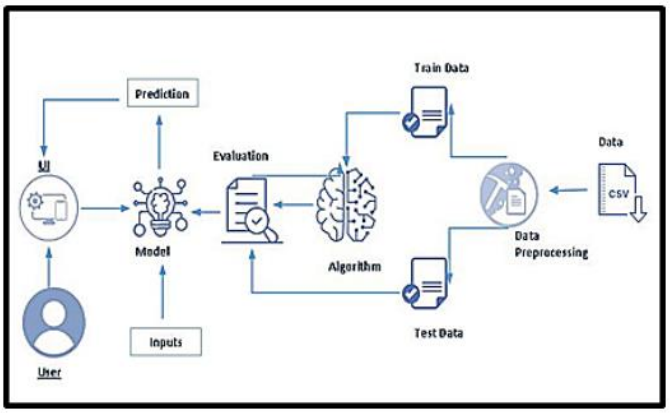
Operative mortality rates have been a topic of great interest among surgeons, patients, lawyers, and health policy administrators. Postoperative respiratory complications are the most common fatality following any type of thoracic surgery. The exact incidence is most contingent upon the preoperative health and lung function of the patient, and we would like to explore and understand how those conditions can drive these complications. One particular metric that has been used to quantify mortality rates in the past has been the thirty-day mortality rate. This metric, however, may not be entirely comprehensive because many patients die shortly after this time period or become very weak, having to be taken to another facility before passing away there. As a result, many of these deaths are severely underreported.

2.2 PROPOSED SOLUTION

The scope of our project is to examine the mortality of patients within a full year after the surgery. More specifically, we are examining the underlying health factors of patients that could potentially be a powerful predictor for surgically related deaths. As mentioned, our feature set includes both continuous and classification data regarding to the patient's health conditions at the time of the surgery. Each patient has 16 variables associated with them Some of the continuous data includes a patients forced vital capacity, the maximum volume their lungs exhaled, size of original tumor, and age at surgery. In addition, we have several classification features such as presence of pain before surgery, haemoptysis before surgery, cough before surgery, whether the patient is a smoker, whether the patient has asthma, and a few others. The classification predicts whether the patient survived the following year long period.

3. THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM



3.2 HARDWARW/SOFTWARE DESIGNING

To complete this project, you must require following software's, concepts, and packages

1. Anaconda navigator and pharm:

a. download anaconda navigator

2. Python packages:

a. Open anaconda prompt as administrator

b. Type "pip install numpy" and click enter.

c. Type "pip install pandas" and click enter.

d. Type "pip install scikit-learn" and click enter.

e. Type"pip install matplotlib" and click enter.

f. Type"pip install scipy" and click enter.

g. Type"pip install pickle-mixin" and click enter.

h. Type"pip install seaborn" and click enter.

i. Type "pip install Flask" and click enter.

4. EXPERIMENTAL INVESTIGATIONS

1. Know fundamental concepts and techniques used for machine learning.

2. Gain a broad understanding about data.

3. Have knowledge on pre-processing the data/transformation techniques on outlier and some visualization concepts.

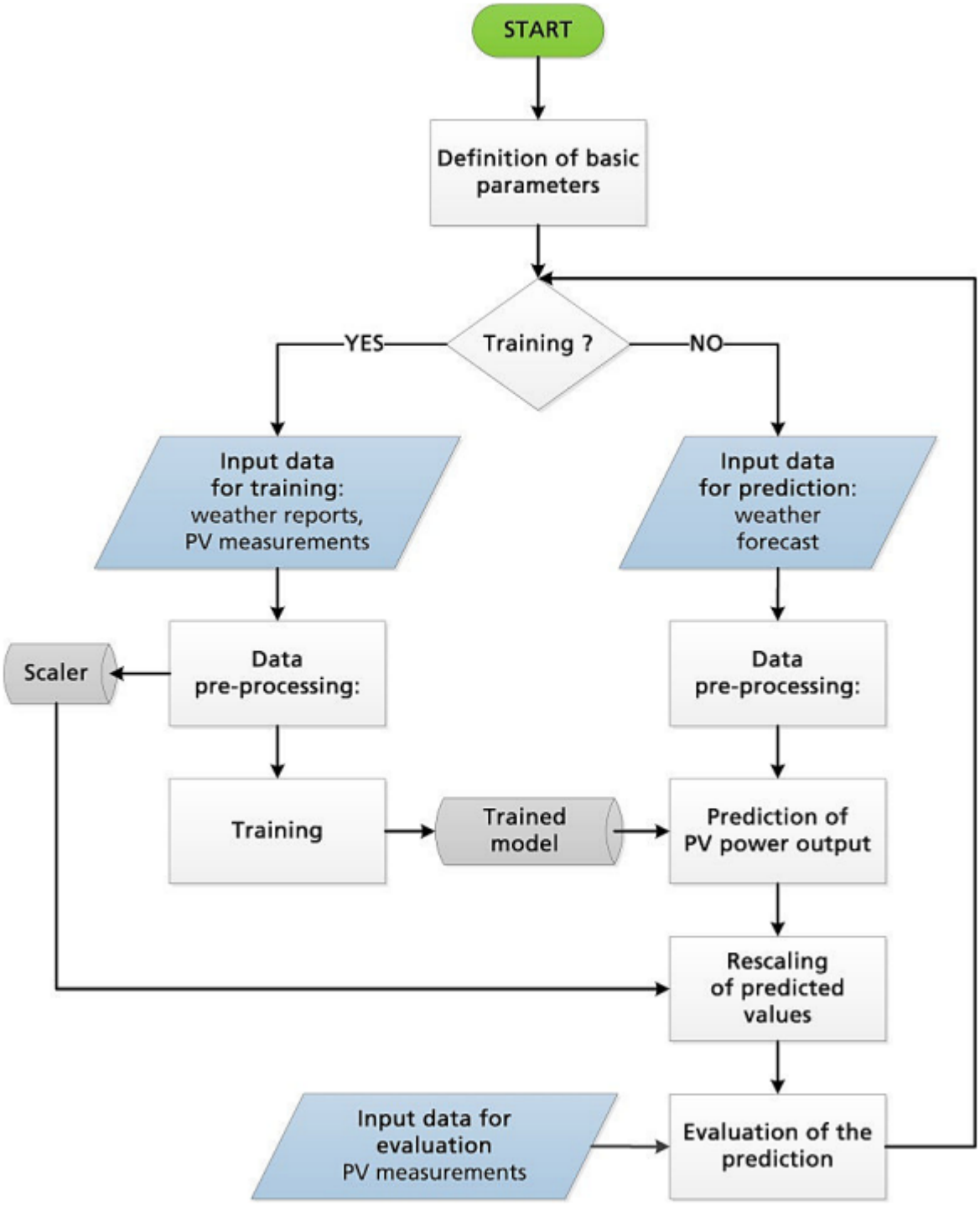
Project Flow:

1. User interacts with the UI to enter the input.
2. Entered input is analyzed by the model which is integrated.
3. Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

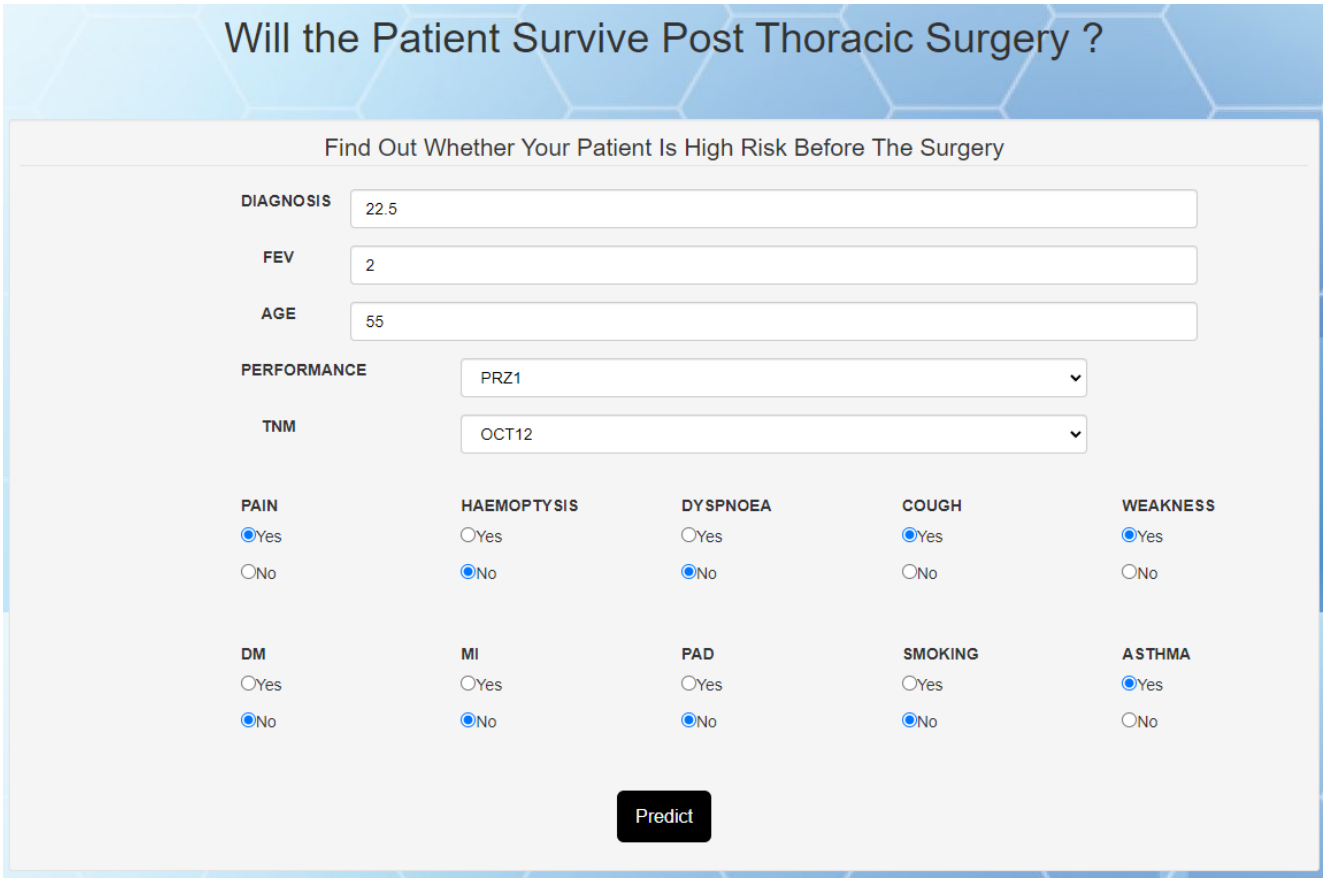
1. Data collection
    a. Collect the dataset or create the dataset
2. Visualizing and analyzing data
    a. Univariate analysis
    b. Bivariate analysis
    c. Multivariate analysis
    d. Descriptive analysis
3. Data pre-processing
    a. Checking for null values
    b. Handling outlier
    c. Handling categorical data
    d. Splitting data into train and test

5. FLOWCHART


START
Definition of basic parameters
Training ?
YES
NO
Input data for training: weather reports, PV measurements
Input data for prediction: weather forecast
Scaler
Data pre-processing:
Data pre-processing:
Training
Trained model
Prediction of PV power output
Rescaling of predicted values
Input data for evaluation PV measurements
Evaluation of the prediction

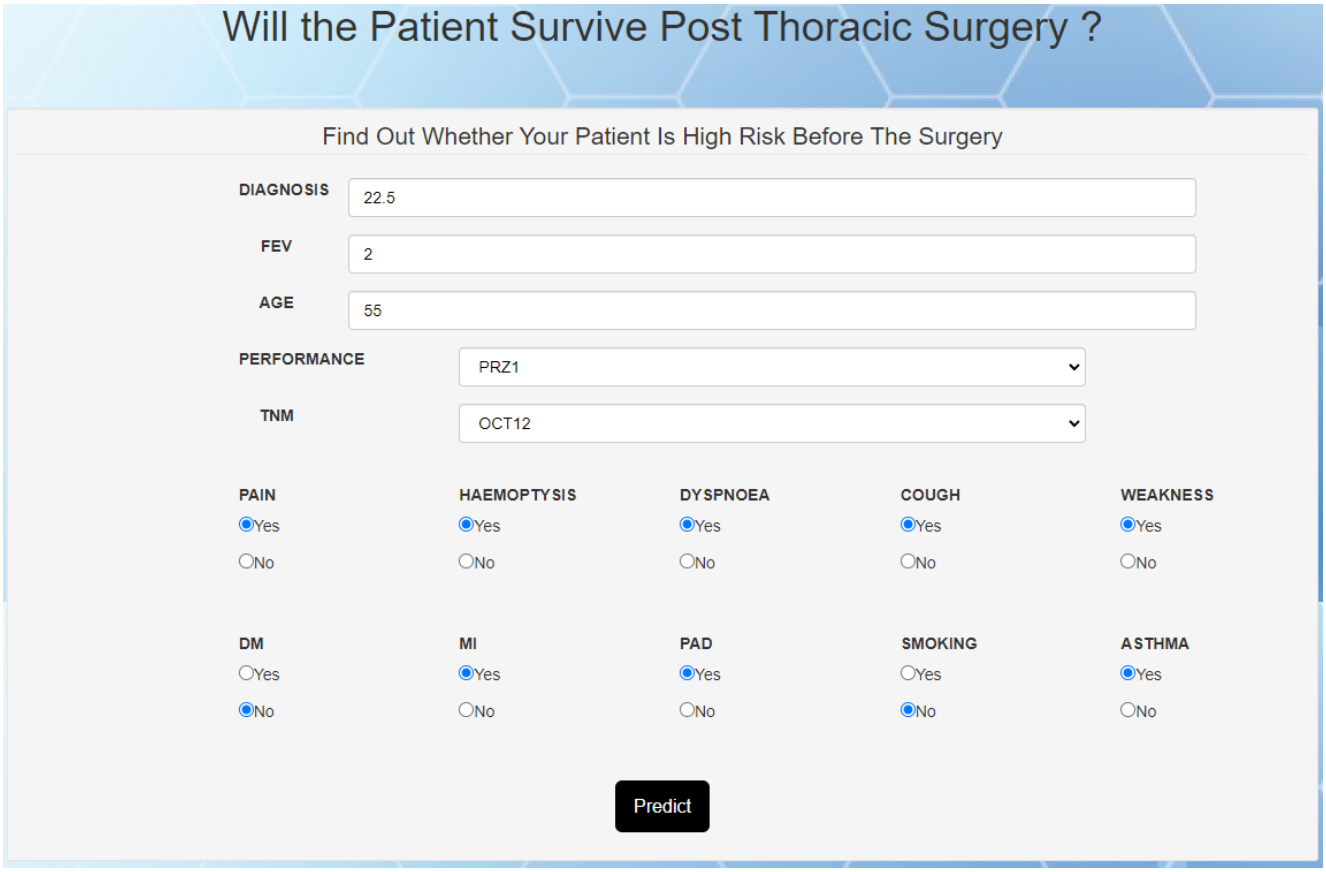
6. RESULT

INPUT 1



OUTPUT 1

INPUT 2



OUTPUT 2

7. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Easy and fast to perform
- ML include lack of interpretability, low quality and volume of relevant clinical data

DISADVANTAGES

- Time consuming
- Skill demanding risks of neurological injury

8. APPLICATIONS

Five key applications of ML to cardiac surgery include diagnostics, surgical skill assessment, postoperative prognostication, augmenting intraoperative performance and accelerating transitional research.

9. CONCLUSION

Lung cancer is one of the challenging problems in medical field due to structure of cancer cells. Therefore, the proper medication has to be given to the patient for increasing the survival chances of the patient. Once the cancer is detected, the Thoracic Surgery is one of the best treatment options for the diagnosis of Lung Cancer. The project involves the analysis of the patient's dataset who underwent Thoracic Surgery, and an attempt is made to model a classifier that will predict the survival of the patient post the surgery. The dataset will be trained using four Supervised the Algorithms of Machine Learning that are Decision Tree, Random Forest, KNN, Xgboost. Among the four algorithms used, its observed that all the algorithms give the highest accuracy of 91% compared to the other algorithms data in the future to analyse the system.

10. FUTURE SCOPE

Ultimately, we were able to improve our results by averaging a series of identical and independently distributed trees, which we would like to contrast with boosting, in which the trees would be grown in an adaptive manner specific to the bias (not I.I.D.). We would like to recursively train on the residuals of each misclassification. A next possible step would be to implement the following algorithm (bumping):

1. Bootstrap n models (with replacement, forcing even ratios), where number of models = number of features.

2. Train n models, with initially one feature per model.

3. Test all n models on original data set. Pick the model with lowest error on original data set and define a new residual data set on all misclassified examples.

4.Train your next n models on the residual (i.e., boosting) - but NO averaging at this point.

5. Test on the very original data set and pick the best one. Continue process repeatedly.

Hopefully this will further reduce our variance. In addition, we calculated the optimal feature set as shown above, so it would be interesting to compare different results for all of our implementations if we use only those specific features

11. BIBILOGRAPHY

[Wroclaw University Study] Creators: Marek Lubicz (1), Konrad Pawelczyk (2), Adam Rzechonek (2), Jerzy Kolodziej (2)

– (1) Wroclaw University of Technology, wybrzeze Wyspianskiego 27, 50-370, Wroclaw, Poland.

– (2) Wroclaw Medical University, wybrzeze L. Pasteura 1, 50-367 Wroclaw, Poland.

Boosted SVM for extracting rules from imbalanced data in application to prediction of the postoperative life expectancy in the lung cancer patients. Applied Soft Computing.

APPENDIX: -

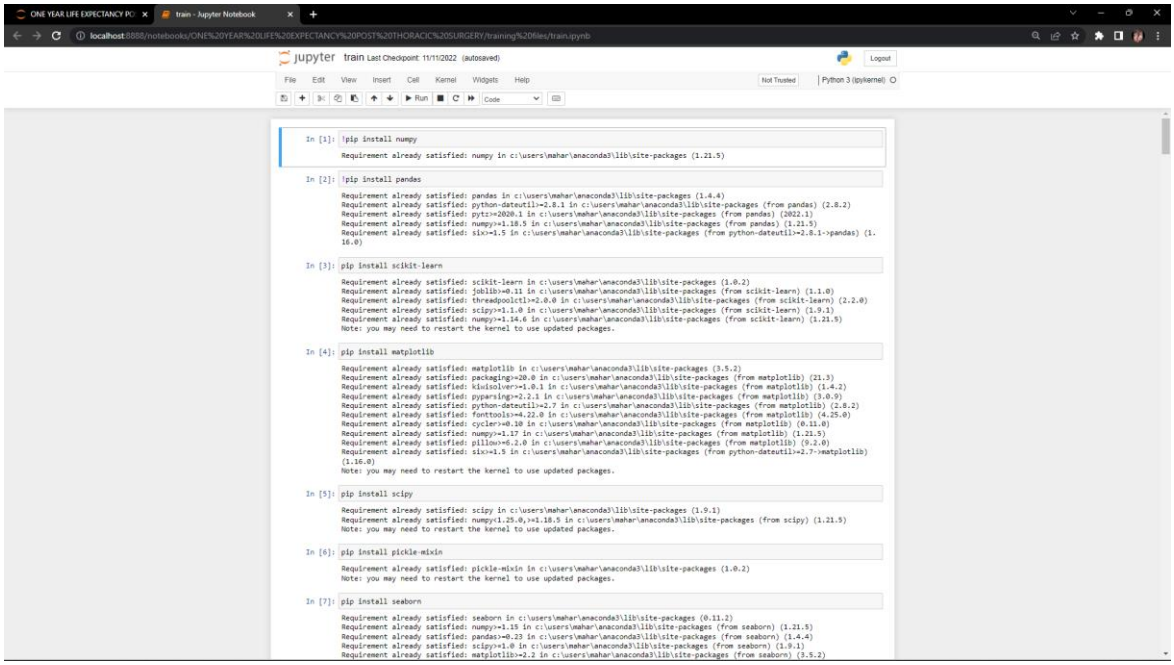
App.py

```
import numpy as np
import pandas as pd
import os
import joblib
from flask import Flask, jsonify, request, render_template, url_for, redirect, Markup


app = Flask(__name__)
joblib_file = "model.pkl"
model = joblib.load(joblib_file)



@app.route('/')
def index():
    return render_template('index.html')
```

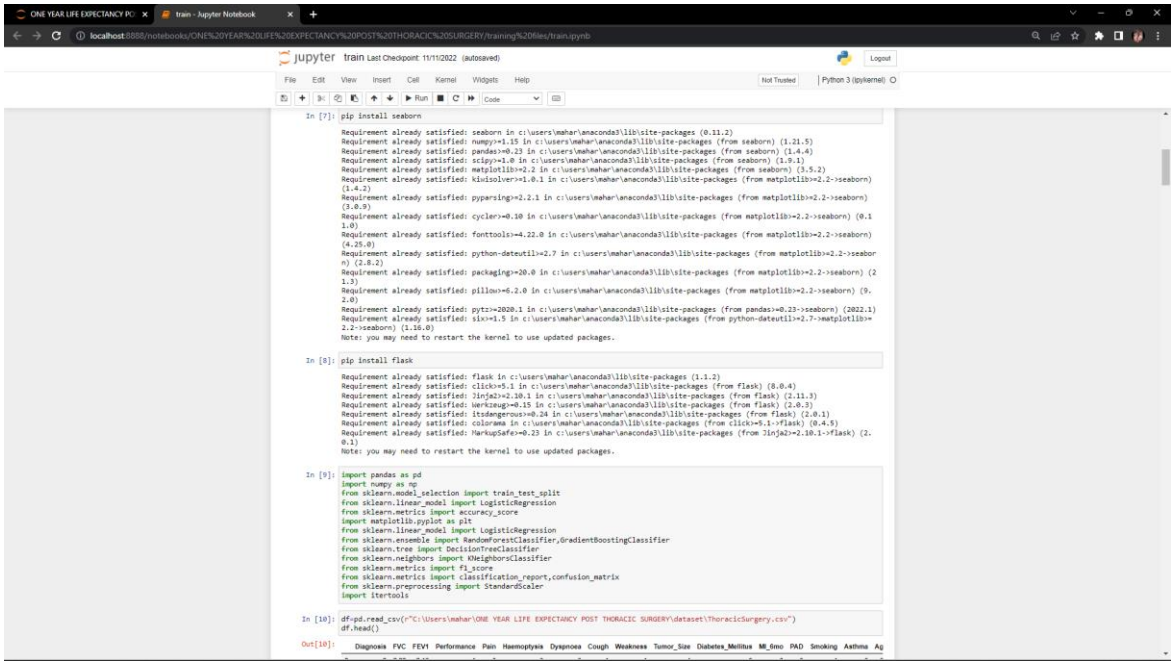
```
@app.route("/form", methods=['GET','POST'])
def getform():
    if request.method == "GET":
        return (render_template("form.html"))


    if request.method == 'POST':
        if 'submit-button' in request.form:
            diagnosis = request.form["diagnosis"]
            fev = request.form["fev"]
            age = request.form["age"]
            performance = request.form["performance"]
            tnm = request.form["tnm"]
            hae = request.form['hae']
            pain = request.form["pain"]
            dys = request.form["dys"]
            cough = request.form["cough"]
            weakness = request.form["weakness"]
            dm = request.form["dm"]
            mi = request.form["mi"]
            pad = request.form["pad"]
            smoking = request.form["smoking"]
            asthma = request.form["asthma"]
            total = [[diagnosis,fev,age,performance,tnm,hae,pain,dys,cough,weakness,dm,mi,pad,smoking,asthma]]
            #prediction = model.predict(total)


            #input_variables = pd.DataFrame([[performance, dys, cough, tnm, dm]], columns=['Performance', 'Dyspnoea', 'Cough', 'TNM', 'DM'], dtype=float)
```

```
        prediction = model.predict(total)[0]

        if int(prediction) == 1:
            prediction = "Patient is at High Risk"


        else:
            prediction = "Patient is Not at Risk"


        return render_template("result.html", prediction = prediction)

    return render_template("result.html")

if __name__=="__main__":
    app.run(debug=False)
```

train.ipynb

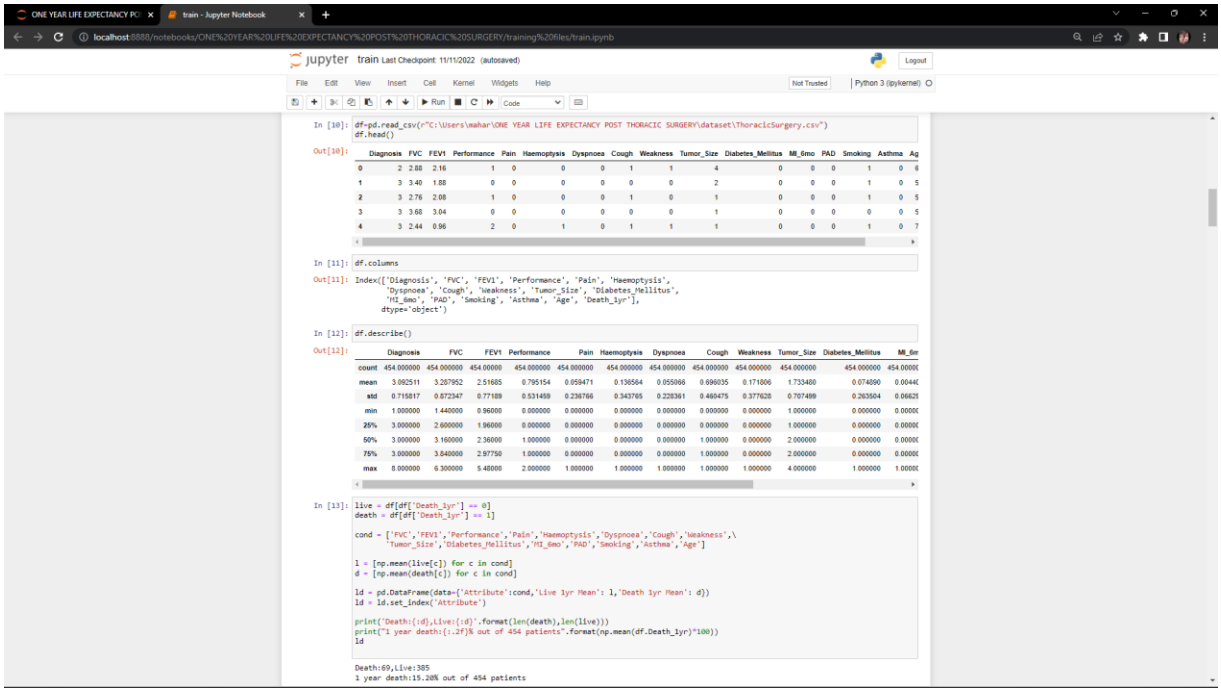


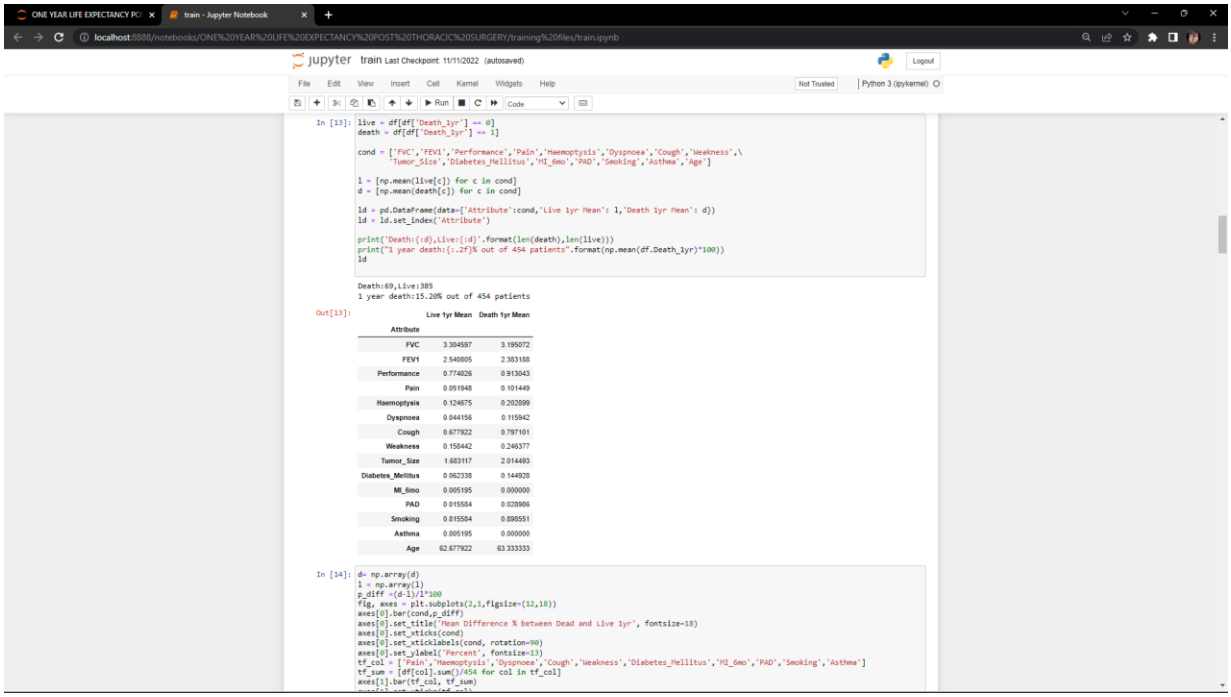
```
In [1]: !pip install numpy
Requirement already satisfied: numpy in c:\users\mahar\anaconda3\lib\site-packages (1.21.5)

In [2]: !pip install pandas
Requirement already satisfied: pandas in c:\users\mahar\anaconda3\lib\site-packages (1.4.4)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\mahar\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\mahar\anaconda3\lib\site-packages (from pandas) (2022.1)
Requirement already satisfied: numpy>=1.18.5 in c:\users\mahar\anaconda3\lib\site-packages (from pandas) (1.21.5)
Requirement already satisfied: six>=1.5 in c:\users\mahar\anaconda3\lib\site-packages (from python-dateutil>=2.8.1->pandas) (1.
16.0)

In [3]: pip install scikit-learn
Requirement already satisfied: scikit-learn in c:\users\mahar\anaconda3\lib\site-packages (1.0.2)
Requirement already satisfied: joblib>=0.11 in c:\users\mahar\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\mahar\anaconda3\lib\site-packages (from scikit-learn) (2.2.0)
Requirement already satisfied: scipy>=1.1.0 in c:\users\mahar\anaconda3\lib\site-packages (from scikit-learn) (1.9.1)
Requirement already satisfied: numpy>=1.14.6 in c:\users\mahar\anaconda3\lib\site-packages (from scikit-learn) (1.21.5)
Note: you may need to restart the kernel to use updated packages.

In [4]: pip install matplotlib
Requirement already satisfied: matplotlib in c:\users\mahar\anaconda3\lib\site-packages (3.5.2)
Requirement already satisfied: packaging>=20.0 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (21.3)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (1.4.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: cycler>=0.10 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: numpy>=1.17 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (1.21.5)
Requirement already satisfied: pillow>=6.2.0 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib) (9.2.0)
Requirement already satisfied: six>=1.5 in c:\users\mahar\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib)
(1.16.0)
Note: you may need to restart the kernel to use updated packages.

In [5]: pip install scipy
Requirement already satisfied: scipy in c:\users\mahar\anaconda3\lib\site-packages (1.9.1)
Requirement already satisfied: numpy<1.25.0,>=1.18.5 in c:\users\mahar\anaconda3\lib\site-packages (from scipy) (1.21.5)
Note: you may need to restart the kernel to use updated packages.

In [6]: pip install pickle-mixin
Requirement already satisfied: pickle-mixin in c:\users\mahar\anaconda3\lib\site-packages (1.0.2)
Note: you may need to restart the kernel to use updated packages.

In [7]: pip install seaborn
Requirement already satisfied: seaborn in c:\users\mahar\anaconda3\lib\site-packages (0.11.2)
Requirement already satisfied: numpy>=1.15 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (1.21.5)
Requirement already satisfied: pandas>=0.23 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (1.4.4)
Requirement already satisfied: scipy>=1.0 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (1.9.1)
Requirement already satisfied: matplotlib>=2.2 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (3.5.2)
```

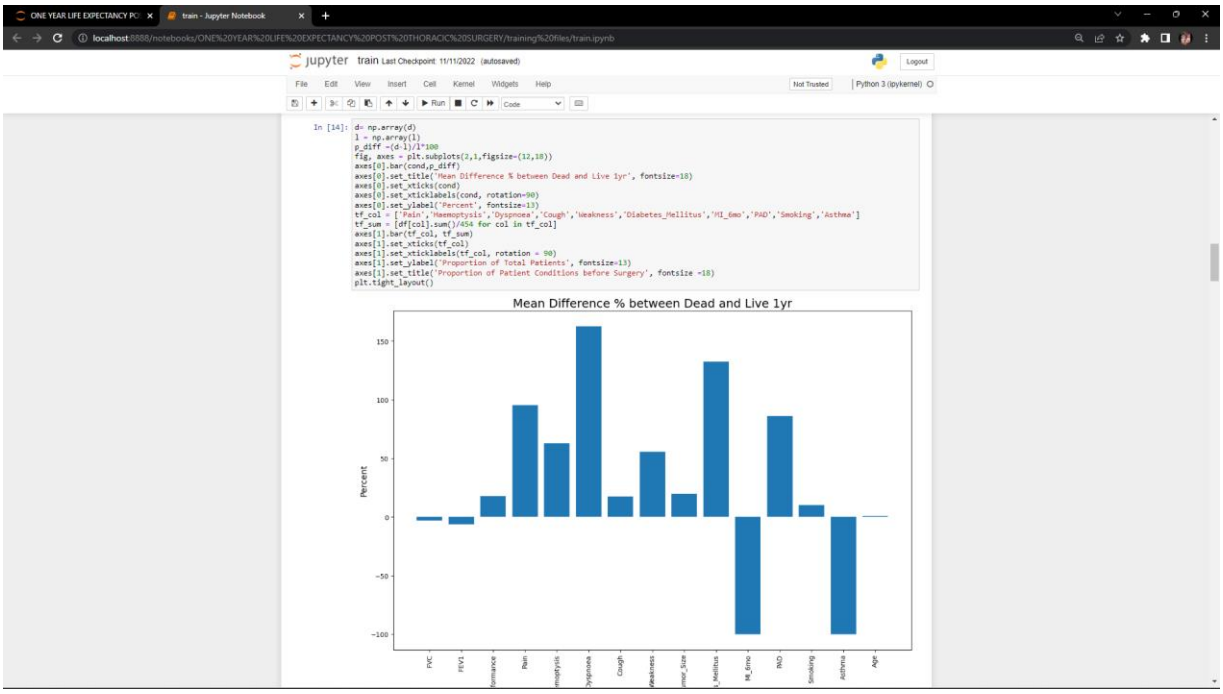


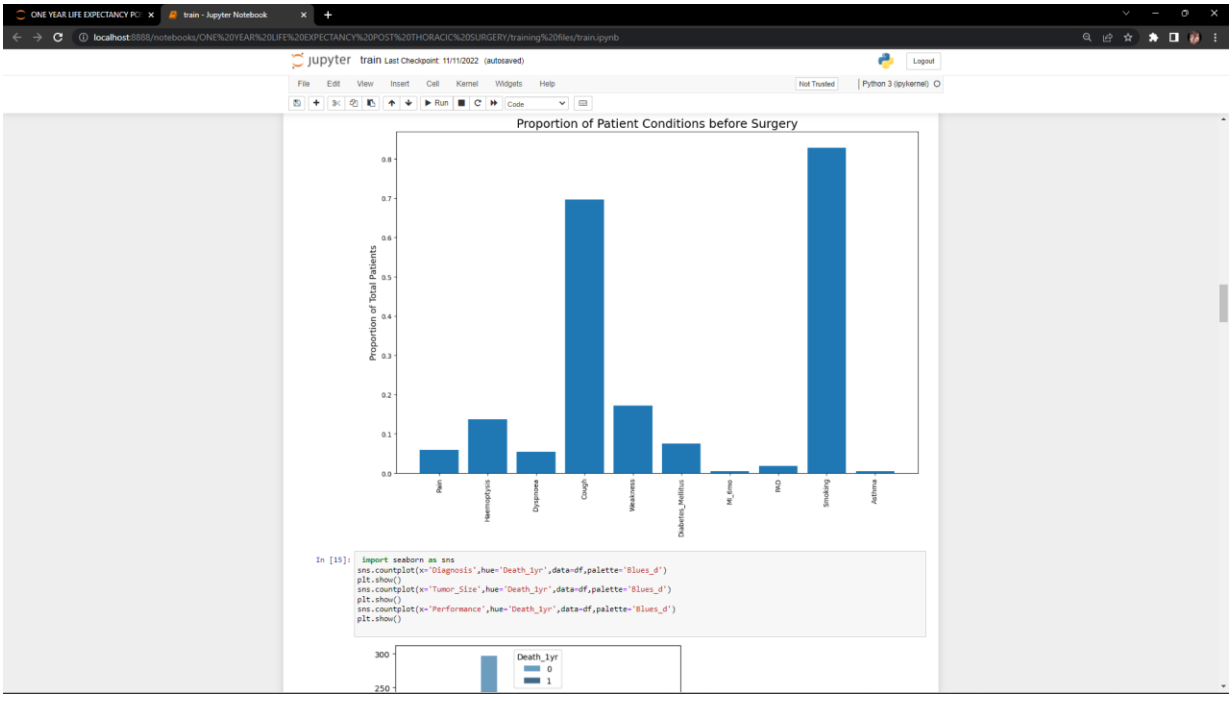
```
In [7]: pip install seaborn
Requirement already satisfied: seaborn in c:\users\mahar\anaconda3\lib\site-packages (0.11.2)
Requirement already satisfied: numpy>=1.15 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (1.21.5)
Requirement already satisfied: pandas>=0.23 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (1.4.4)
Requirement already satisfied: scipy>=1.0 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (1.9.1)
Requirement already satisfied: matplotlib>=2.2 in c:\users\mahar\anaconda3\lib\site-packages (from seaborn) (3.5.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn)
(1.4.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn)
(3.0.9)
Requirement already satisfied: cycler>=0.10 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (0.1
1.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn)
(4.25.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seabor
n) (2.8.2)
Requirement already satisfied: packaging>=20.0 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (2
1.3)
Requirement already satisfied: pillow>=6.2.0 in c:\users\mahar\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (9.
2.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\mahar\anaconda3\lib\site-packages (from pandas>=0.23->seaborn) (2022.1)
Requirement already satisfied: six>=1.5 in c:\users\mahar\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib>=
2.2->seaborn) (1.16.0)
Note: you may need to restart the kernel to use updated packages.

In [8]: pip install flask
Requirement already satisfied: flask in c:\users\mahar\anaconda3\lib\site-packages (1.1.2)
Requirement already satisfied: click>=5.1 in c:\users\mahar\anaconda3\lib\site-packages (from flask) (8.0.4)
Requirement already satisfied: Jinja2>=2.10.1 in c:\users\mahar\anaconda3\lib\site-packages (from flask) (2.11.3)
Requirement already satisfied: Werkzeug>=0.15 in c:\users\mahar\anaconda3\lib\site-packages (from flask) (2.0.3)
Requirement already satisfied: itsdangerous>=0.24 in c:\users\mahar\anaconda3\lib\site-packages (from flask) (2.0.1)
Requirement already satisfied: colorama in c:\users\mahar\anaconda3\lib\site-packages (from click>=5.1->flask) (0.4.5)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\mahar\anaconda3\lib\site-packages (from Jinja2>=2.10.1->flask) (2.
0.1)
Note: you may need to restart the kernel to use updated packages.

In [9]: import pandas as pd
        import numpy as np
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import accuracy_score
        import matplotlib.pyplot as plt
        from sklearn.linear_model import LogisticRegression
        from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.metrics import f1_score
        from sklearn.metrics import classification_report,confusion_matrix
        from sklearn.preprocessing import StandardScaler
        import itertools

In [10]: df=pd.read_csv(r"C:\Users\mahar\ONE YEAR LIFE EXPECTANCY POST THORACIC SURGERY\dataset\ThoracicSurgery.csv")
         df.head()

Out[10]:
```

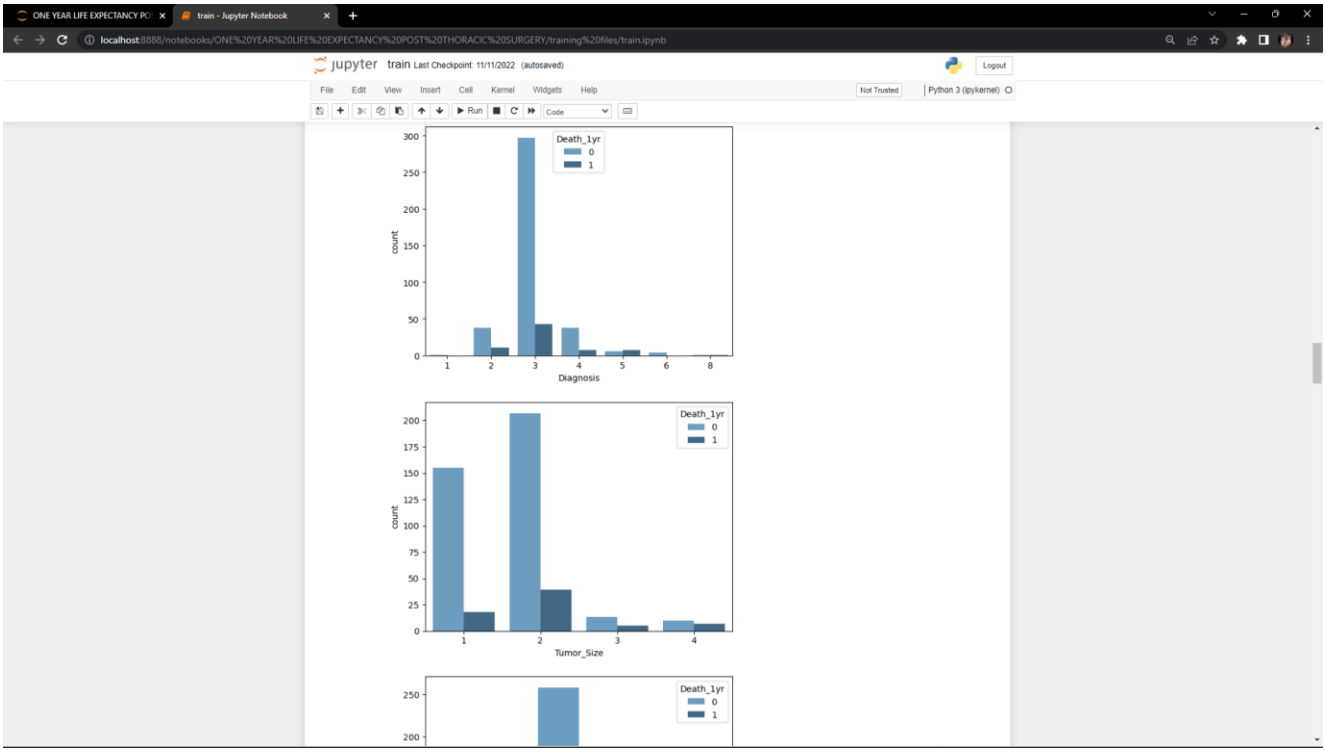
	Diagnosis	FVC	FEV1	Performance	Pain	Haemoptysis	Dyspnoea	Cough	Weakness	Tumor_Size	Diabetes_Mellitus	MI_6mo	PAD	Smoking	Asthma	Ag

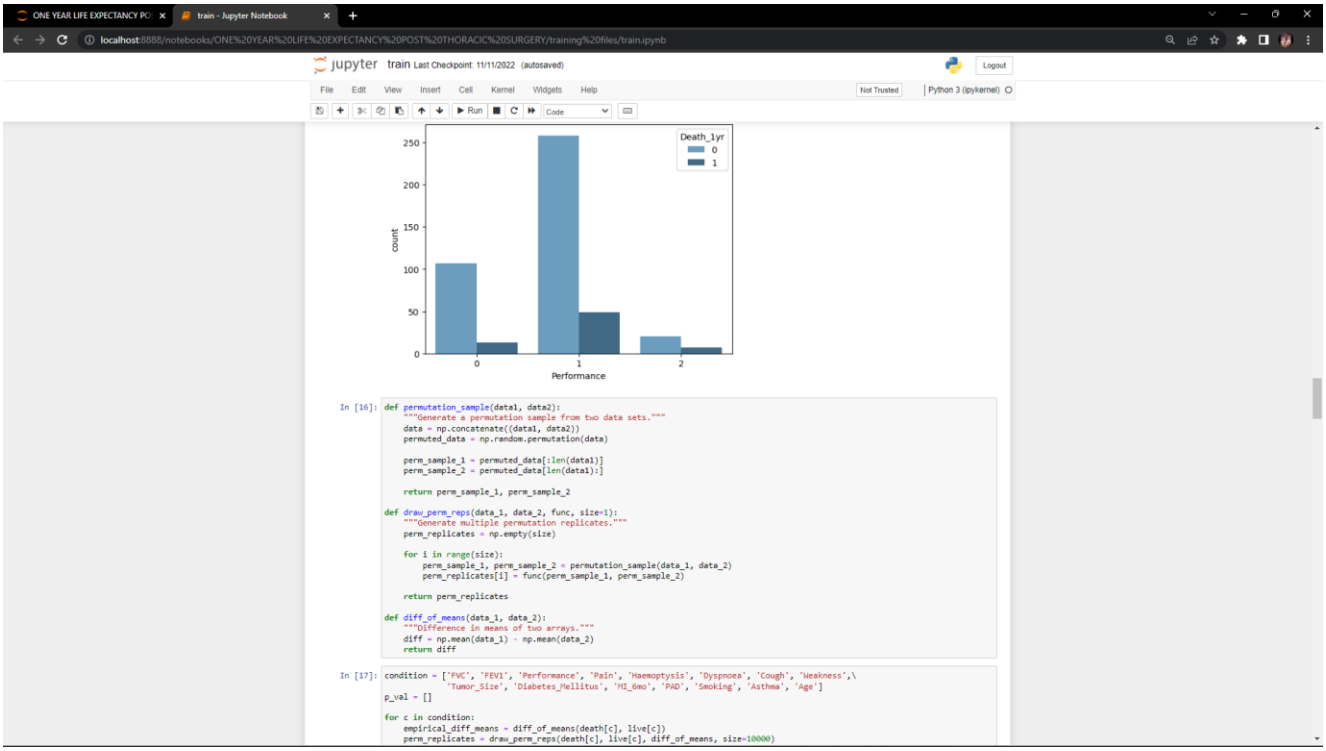

ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [10]:
df=pd.read_csv(r"C:\Users\mahar\ONE YEAR LIFE EXPECTANCY POST THORACIC SURGERY\dataset\ThoracicSurgery.csv")
df.head()
Out[10]:
Diagnosis FVC FEV1 Performance Pain Haemoptysis Dyspnoea Cough Weakness Tumor_Size Diabetes_Mellitus MI_6mo PAD Smoking Asthma Ag
0 2 2.88 2.16 1 0 0 0 1 1 4 0 0 0 1 0 6
1 3 3.40 1.88 0 0 0 0 0 0 2 0 0 0 1 0 5
2 3 2.76 2.08 1 0 0 0 1 0 1 0 0 0 1 0 5
3 3 3.68 3.04 0 0 0 0 0 0 1 0 0 0 0 0 5
4 3 2.44 0.96 2 0 1 0 1 1 1 0 0 0 1 0 7
In [11]:
df.columns
Out[11]:
Index(['Diagnosis', 'FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis',
'Dyspnoea', 'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus',
'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age', 'Death_1yr'],
dtype='object')
In [12]:
df.describe()
Out[12]:
Diagnosis FVC FEV1 Performance Pain Haemoptysis Dyspnoea Cough Weakness Tumor_Size Diabetes_Mellitus MI_6m
count 454.000000 454.000000 454.000000 454.000000 454.000000 454.000000 454.000000 454.000000 454.000000 454.000000 454.000000 454.0000
mean 3.092511 3.287952 2.51685 0.795154 0.059471 0.136564 0.055066 0.696035 0.171806 1.733480 0.074890 0.0044
std 0.715817 0.872347 0.77189 0.531459 0.236766 0.343765 0.228361 0.460475 0.377628 0.707499 0.263504 0.0662
min 1.000000 1.440000 0.96000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.0000
25% 3.000000 2.600000 1.96000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.0000
50% 3.000000 3.160000 2.36000 1.000000 0.000000 0.000000 0.000000 1.000000 0.000000 2.000000 0.000000 0.0000
75% 3.000000 3.840000 2.97750 1.000000 0.000000 0.000000 0.000000 1.000000 0.000000 2.000000 0.000000 0.0000
max 8.000000 6.300000 5.48000 2.000000 1.000000 1.000000 1.000000 1.000000 1.000000 4.000000 1.000000 1.0000
In [13]:
live = df[df['Death_1yr'] == 0]
death = df[df['Death_1yr'] == 1]
cond = ['FVC','FEV1','Performance','Pain','Haemoptysis','Dyspnoea','Cough','Weakness',\
'Tumor_Size','Diabetes_Mellitus','MI_6mo','PAD','Smoking','Asthma','Age']
l = [np.mean(live[c]) for c in cond]
d = [np.mean(death[c]) for c in cond]
ld = pd.DataFrame(data={'Attribute':cond,'Live 1yr Mean': l,'Death 1yr Mean': d})
ld = ld.set_index('Attribute')
print('Death:{:d},Live:{:d}'.format(len(death),len(live)))
print("1 year death:{:.2f}% out of 454 patients".format(np.mean(df.Death_1yr)*100))
ld
Death:69,Live:385
1 year death:15.20% out of 454 patients

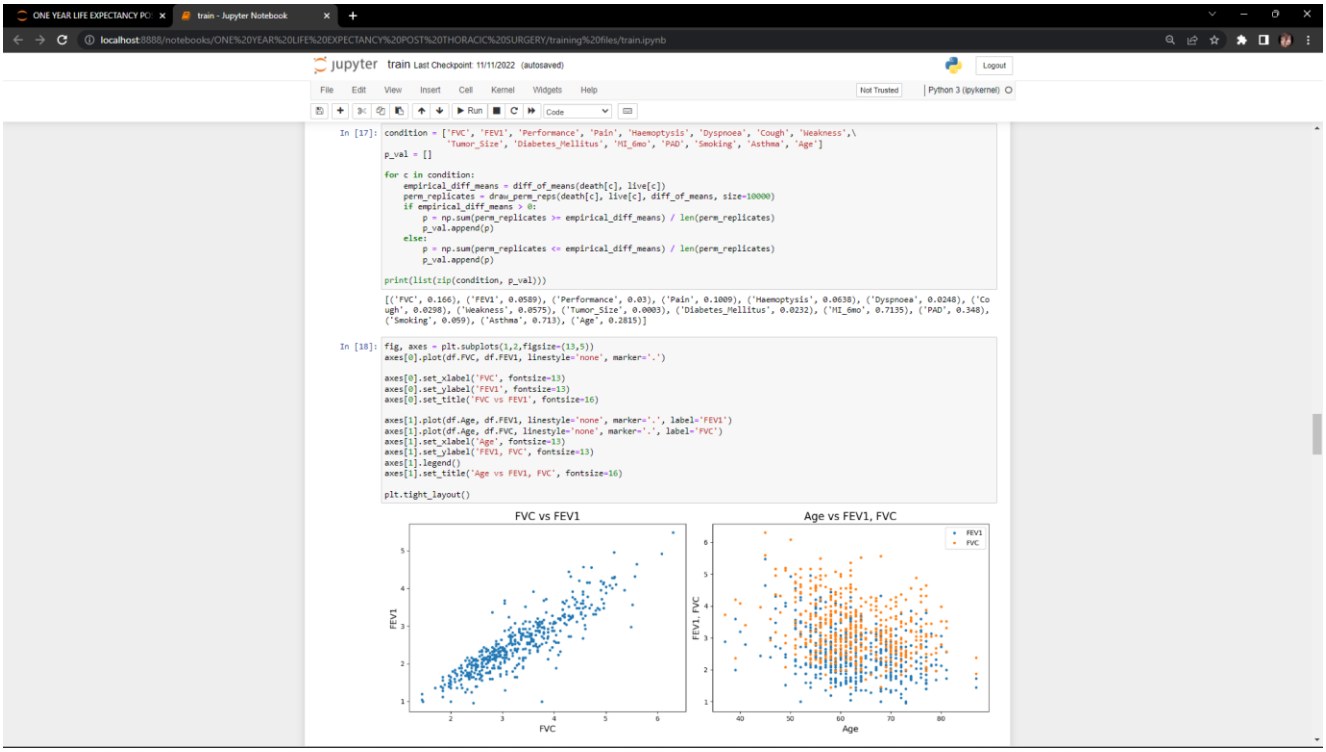

ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [13]:
live = df[df['Death_1yr'] == 0]
death = df[df['Death_1yr'] == 1]
cond = ['FVC','FEV1','Performance','Pain','Haemoptysis','Dyspnoea','Cough','Weakness',\
'Tumor_Size','Diabetes_Mellitus','MI_6mo','PAD','Smoking','Asthma','Age']
l = [np.mean(live[c]) for c in cond]
d = [np.mean(death[c]) for c in cond]
ld = pd.DataFrame(data={'Attribute':cond,'Live 1yr Mean': l,'Death 1yr Mean': d})
ld = ld.set_index('Attribute')
print('Death:{:d},Live:{:d}'.format(len(death),len(live)))
print("1 year death:{:.2f}% out of 454 patients".format(np.mean(df.Death_1yr)*100))
ld
Death:69,Live:385
1 year death:15.20% out of 454 patients
Out[13]:
Live 1yr Mean Death 1yr Mean
Attribute
FVC 3.304597 3.195072
FEV1 2.540805 2.383188
Performance 0.774026 0.913043
Pain 0.051948 0.101449
Haemoptysis 0.124675 0.202899
Dyspnoea 0.044156 0.115942
Cough 0.677922 0.797101
Weakness 0.150442 0.246377
Tumor_Size 1.683117 2.014493
Diabetes_Mellitus 0.062338 0.144928
MI_6mo 0.005195 0.000000
PAD 0.015584 0.028986
Smoking 0.815584 0.898551
Asthma 0.005195 0.000000
Age 62.677922 63.333333
In [14]:
d= np.array(d)
l = np.array(l)
p_diff =(d-l)/l*100
fig, axes = plt.subplots(2,1,figsize=(12,18))
axes[0].bar(cond,p_diff)
axes[0].set_title('Mean Difference % between Dead and Live 1yr', fontsize=18)
axes[0].set_xticks(cond)
axes[0].set_xticklabels(cond, rotation=90)
axes[0].set_ylabel('Percent', fontsize=13)
tf_col = ['Pain','Haemoptysis','Dyspnoea','Cough','Weakness','Diabetes_Mellitus','MI_6mo','PAD','Smoking','Asthma']
tf_sum = [df[col].sum()/454 for col in tf_col]
axes[1].bar(tf_col, tf_sum)

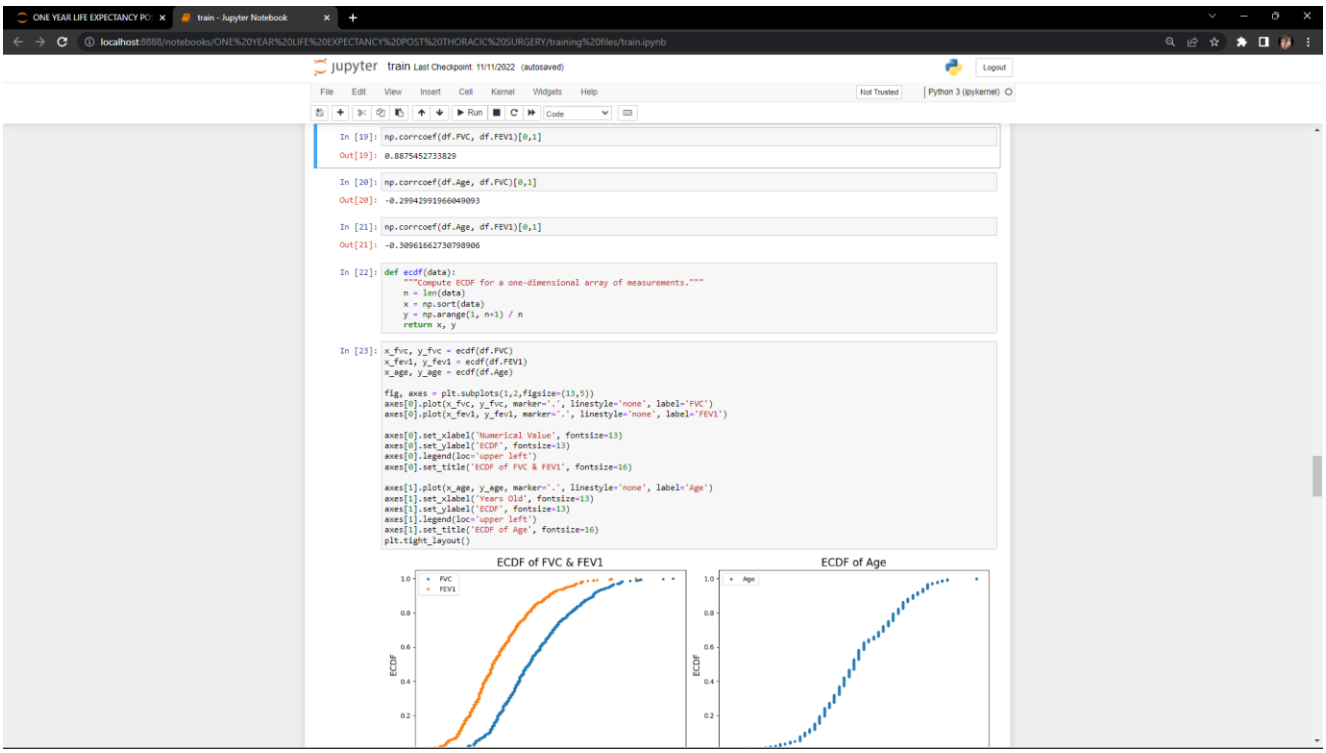

train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [14]:
d= np.array(d)
l = np.array(l)
p_diff =(d-l)/l*100
fig, axes = plt.subplots(2,1,figsize=(12,18))
axes[0].bar(cond,p_diff)
axes[0].set_title('Mean Difference % between Dead and Live 1yr', fontsize=18)
axes[0].set_xticks(cond)
axes[0].set_xticklabels(cond, rotation=90)
axes[0].set_ylabel('Percent', fontsize=13)
tf_col = ['Pain','Haemoptysis','Dyspnoea','Cough','Weakness','Diabetes_Mellitus','MI_6mo','PAD','Smoking','Asthma']
tf_sum = [df[col].sum()/454 for col in tf_col]
axes[1].bar(tf_col, tf_sum)
axes[1].set_xticks(tf_col)
axes[1].set_xticklabels(tf_col, rotation = 90)
axes[1].set_ylabel('Proportion of Total Patients', fontsize=13)
axes[1].set_title('Proportion of Patient Conditions before Surgery', fontsize =18)
plt.tight_layout()
Mean Difference % between Dead and Live 1yr
Percent
150
100
50
0
−50
−100
PVC
FEV1
Performance
Pain
Haemoptysis
Dyspnoea
Cough
Weakness
Tumor_Size
Diabetes_Mellitus
MI_6mo
PAD
Smoking
Asthma
Age

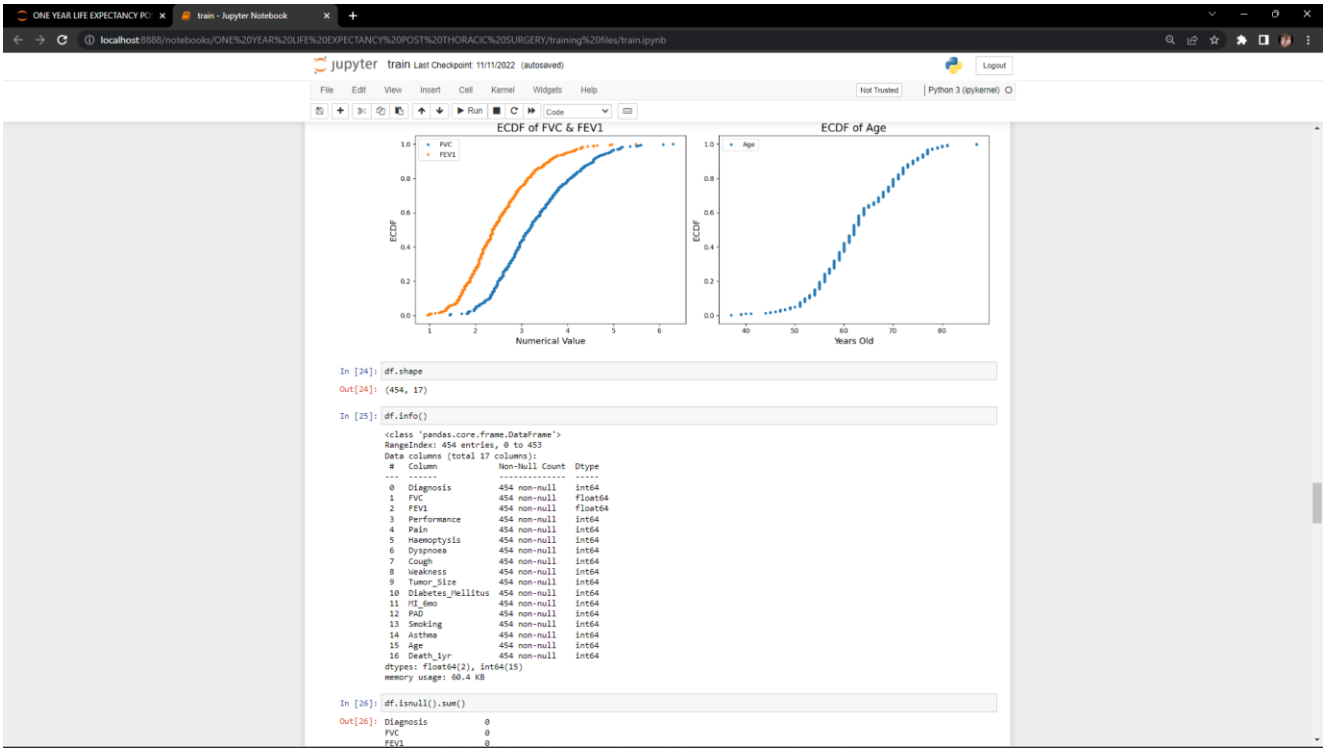

train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Proportion of Patient Conditions before Surgery
Proportion of Total Patients
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Pain
Haemoptysis
Dyspnoea
Cough
Weakness
Diabetes_Mellitus
MI_6mo
PAD
Smoking
Asthma
In [15]:
import seaborn as sns
sns.countplot(x='Diagnosis',hue='Death_1yr',data=df,palette='Blues_d')
plt.show()
sns.countplot(x='Tumor_Size',hue='Death_1yr',data=df,palette='Blues_d')
plt.show()
sns.countplot(x='Performance',hue='Death_1yr',data=df,palette='Blues_d')
plt.show()
300
250
Death_1yr
0
1


ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Death_1yr
0
1
count
Diagnosis
Tumor_Size

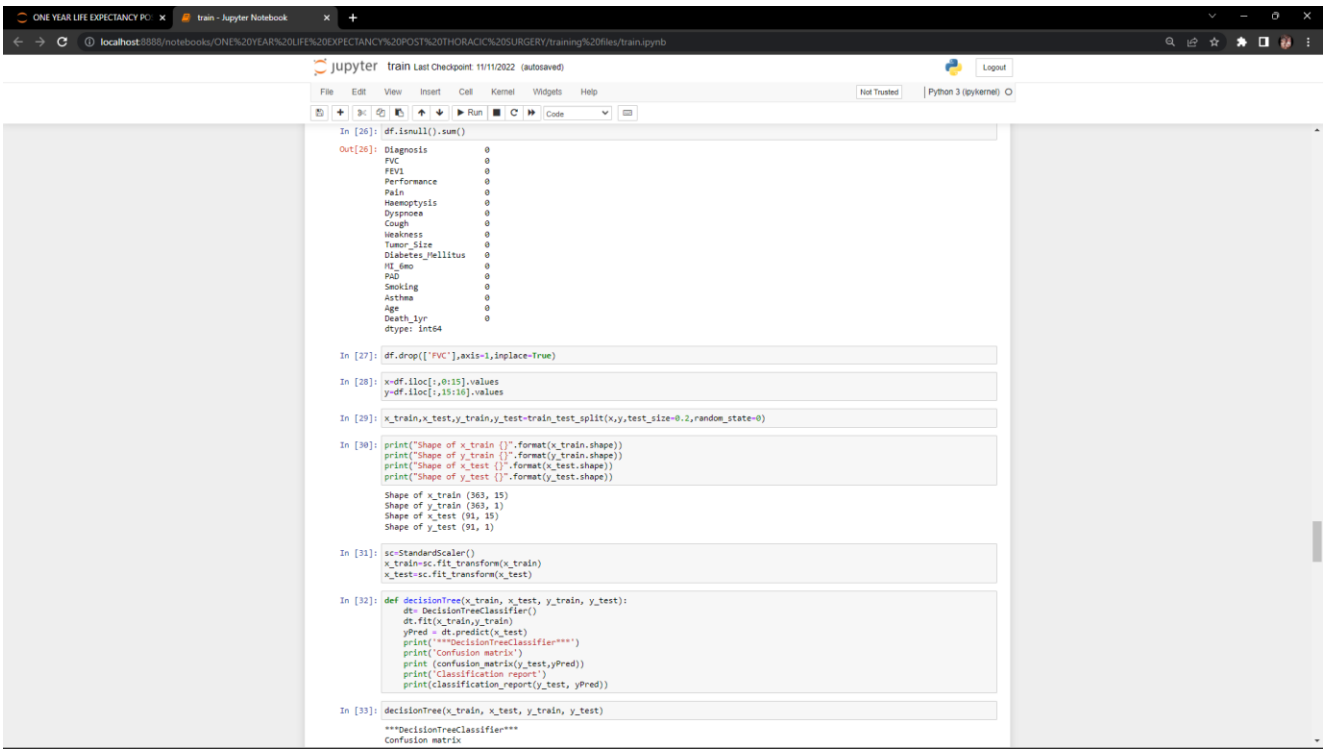

ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Death_1yr
0
1
count
Performance
In [16]:
def permutation_sample(data1, data2):
"""Generate a permutation sample from two data sets."""
data = np.concatenate((data1, data2))
permuted_data = np.random.permutation(data)
perm_sample_1 = permuted_data[:len(data1)]
perm_sample_2 = permuted_data[len(data1):]
return perm_sample_1, perm_sample_2
def draw_perm_reps(data_1, data_2, func, size=1):
"""Generate multiple permutation replicates."""
perm_replicates = np.empty(size)
for i in range(size):
perm_sample_1, perm_sample_2 = permutation_sample(data_1, data_2)
perm_replicates[i] = func(perm_sample_1, perm_sample_2)
return perm_replicates
def diff_of_means(data_1, data_2):
"""Difference in means of two arrays."""
diff = np.mean(data_1) - np.mean(data_2)
return diff
In [17]:
condition = ['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis', 'Dyspnoea', 'Cough', 'Weakness',\
'Tumor_Size', 'Diabetes_Hellitus', 'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age']
p_val = []
for c in condition:
empirical_diff_means = diff_of_means(death[c], live[c])
perm_replicates = draw_perm_reps(death[c], live[c], diff_of_means, size=10000)


ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [17]:
condition = ['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis', 'Dyspnoea', 'Cough', 'Weakness',\
'Tumor_Size', 'Diabetes_Mellitus', 'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age']
p_val = []
for c in condition:
empirical_diff_means = diff_of_means(death[c], live[c])
perm_replicates = draw_perm_reps(death[c], live[c], diff_of_means, size=10000)
if empirical_diff_means > 0:
p = np.sum(perm_replicates >= empirical_diff_means) / len(perm_replicates)
p_val.append(p)
else:
p = np.sum(perm_replicates <= empirical_diff_means) / len(perm_replicates)
p_val.append(p)
print(list(zip(condition, p_val)))
[('FVC', 0.166), ('FEV1', 0.0589), ('Performance', 0.03), ('Pain', 0.1009), ('Haemoptysis', 0.0638), ('Dyspnoea', 0.0248), ('Cough', 0.0298), ('Weakness', 0.0575), ('Tumor_Size', 0.0003), ('Diabetes_Mellitus', 0.0232), ('MI_6mo', 0.7135), ('PAD', 0.348), ('Smoking', 0.059), ('Asthma', 0.713), ('Age', 0.2815)]
In [18]:
fig, axes = plt.subplots(1,2,figsize=(13,5))
axes[0].plot(df.FVC, df.FEV1, linestyle='none', marker='.')
axes[0].set_xlabel('FVC', fontsize=13)
axes[0].set_ylabel('FEV1', fontsize=13)
axes[0].set_title('FVC vs FEV1', fontsize=16)
axes[1].plot(df.Age, df.FEV1, linestyle='none', marker='.', label='FEV1')
axes[1].plot(df.Age, df.FVC, linestyle='none', marker='.', label='FVC')
axes[1].set_xlabel('Age', fontsize=13)
axes[1].set_ylabel('FEV1, FVC', fontsize=13)
axes[1].legend()
axes[1].set_title('Age vs FEV1, FVC', fontsize=16)
plt.tight_layout()
FVC vs FEV1
Age vs FEV1, FVC
FEV1
FVC
FEV1, FVC
Age

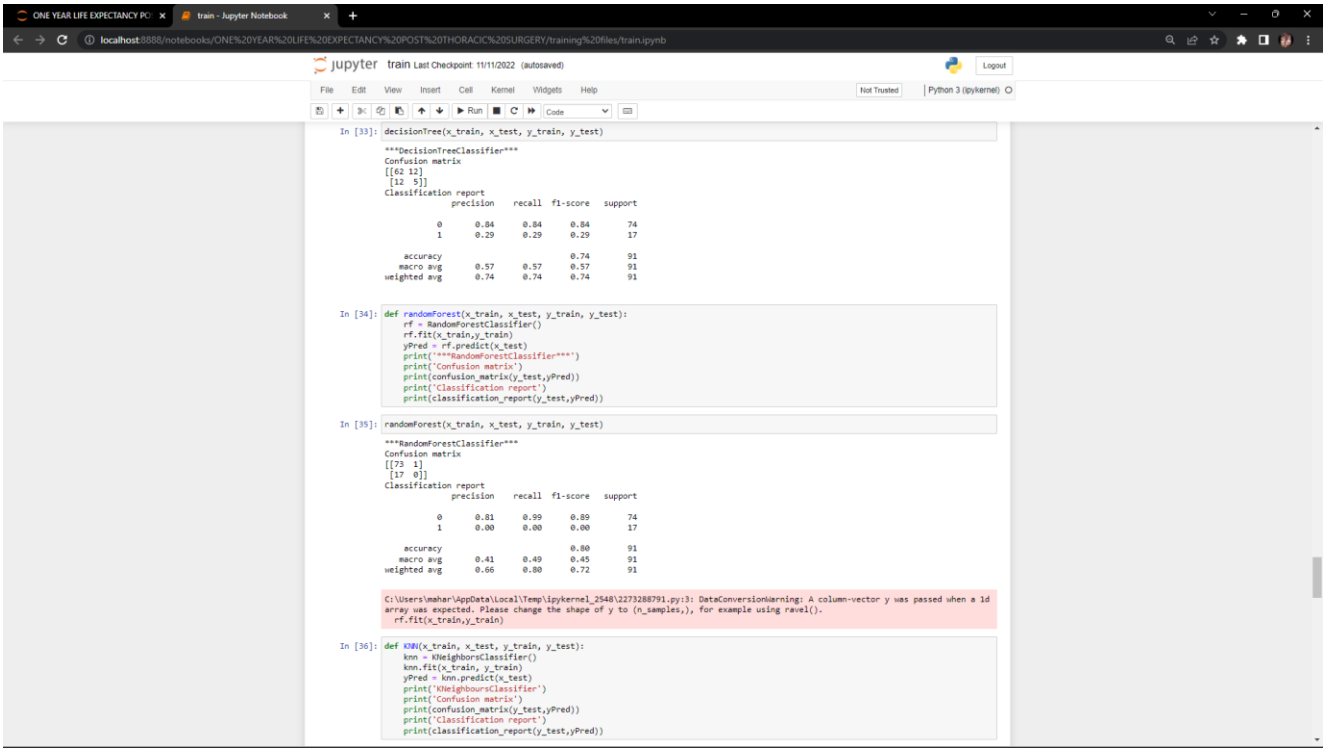

ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [19]:
np.corrcoef(df.FVC, df.FEV1)[0,1]
Out[19]:
0.8875452733829
In [20]:
np.corrcoef(df.Age, df.FVC)[0,1]
Out[20]:
-0.29942991966049093
In [21]:
np.corrcoef(df.Age, df.FEV1)[0,1]
Out[21]:
-0.30961662730798906
In [22]:
def ecdf(data):
"""Compute ECDF for a one-dimensional array of measurements."""
n = len(data)
x = np.sort(data)
y = np.arange(1, n+1) / n
return x, y
In [23]:
x_fvc, y_fvc = ecdf(df.FVC)
x_fev1, y_fev1 = ecdf(df.FEV1)
x_age, y_age = ecdf(df.Age)
fig, axes = plt.subplots(1,2,figsize=(13,5))
axes[0].plot(x_fvc, y_fvc, marker='.', linestyle='none', label='FVC')
axes[0].plot(x_fev1, y_fev1, marker='.', linestyle='none', label='FEV1')
axes[0].set_xlabel('Numerical Value', fontsize=13)
axes[0].set_ylabel('ECDF', fontsize=13)
axes[0].legend(loc='upper left')
axes[0].set_title('ECDF of FVC & FEV1', fontsize=16)
axes[1].plot(x_age, y_age, marker='.', linestyle='none', label='Age')
axes[1].set_xlabel('Years Old', fontsize=13)
axes[1].set_ylabel('ECDF', fontsize=13)
axes[1].legend(loc='upper left')
axes[1].set_title('ECDF of Age', fontsize=16)
plt.tight_layout()
ECDF of FVC & FEV1
ECDF of Age
FVC
FEV1
Age
ECDF

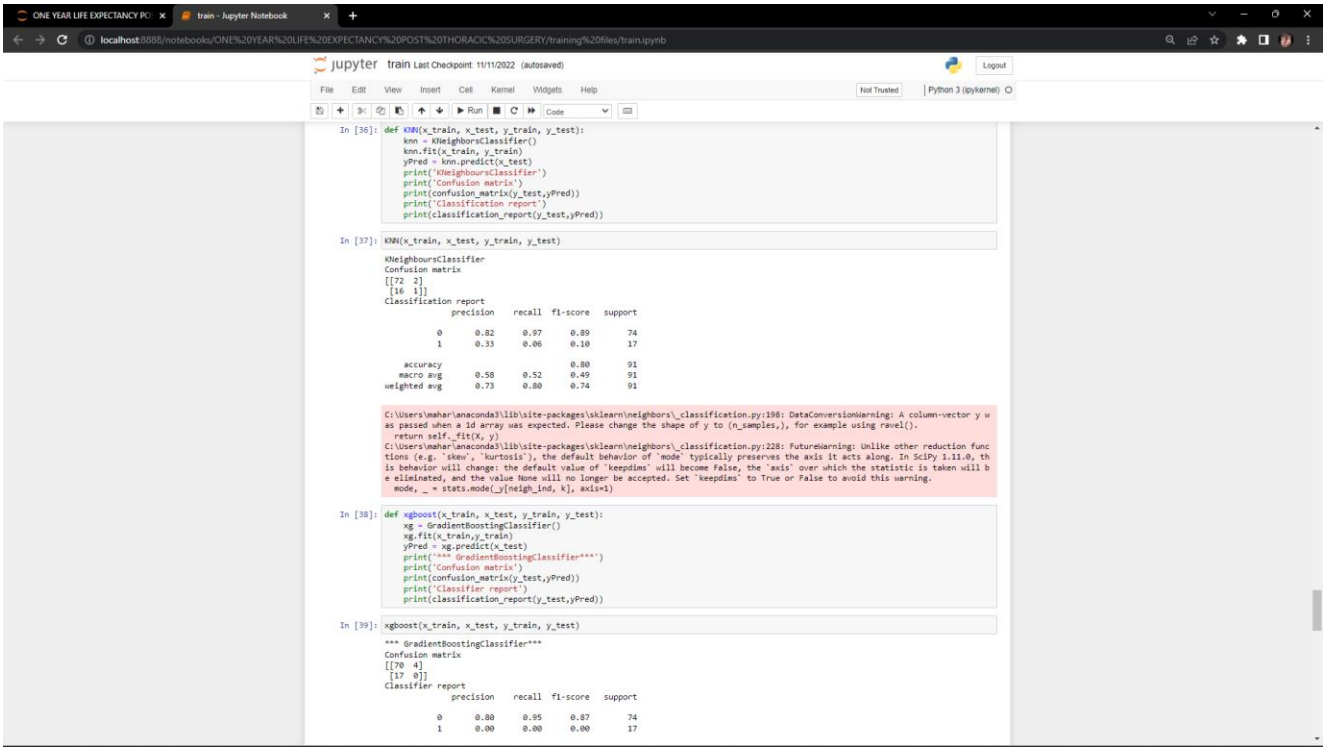


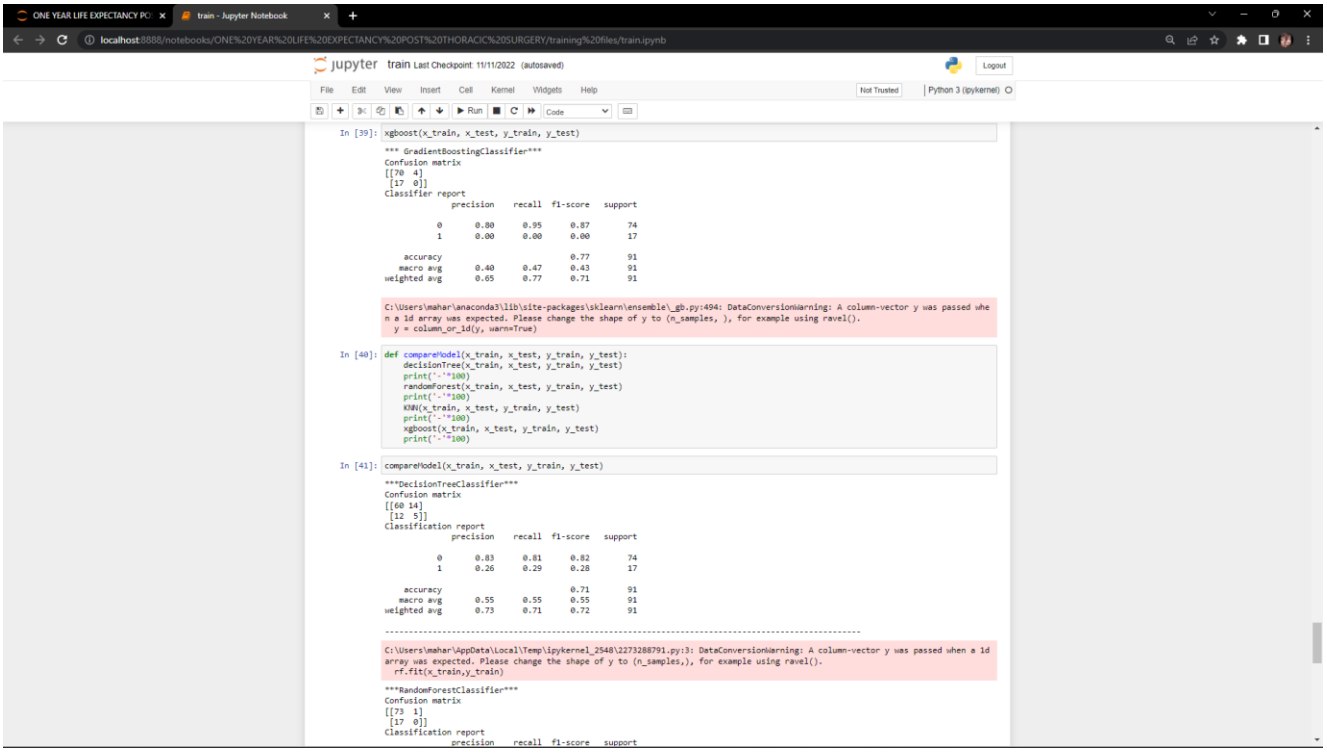

```
In [24]: df.shape
Out[24]: (454, 17)

In [25]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 454 entries, 0 to 453
Data columns (total 17 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Diagnosis           454 non-null    int64
 1   FVC                 454 non-null    float64
 2   FEV1                454 non-null    float64
 3   Performance         454 non-null    int64
 4   Pain                454 non-null    int64
 5   Haemoptysis         454 non-null    int64
 6   Dyspnoea            454 non-null    int64
 7   Cough               454 non-null    int64
 8   Weakness            454 non-null    int64
 9   Tumor_Size          454 non-null    int64
 10  Diabetes_Mellitus   454 non-null    int64
 11  MI_6mo              454 non-null    int64
 12  PAD                 454 non-null    int64
 13  Smoking             454 non-null    int64
 14  Asthma              454 non-null    int64
 15  Age                 454 non-null    int64
 16  Death_1yr           454 non-null    int64
dtypes: float64(2), int64(15)
memory usage: 60.4 KB

In [26]: df.isnull().sum()
Out[26]: Diagnosis           0
         FVC                 0
         FEV1                0
```

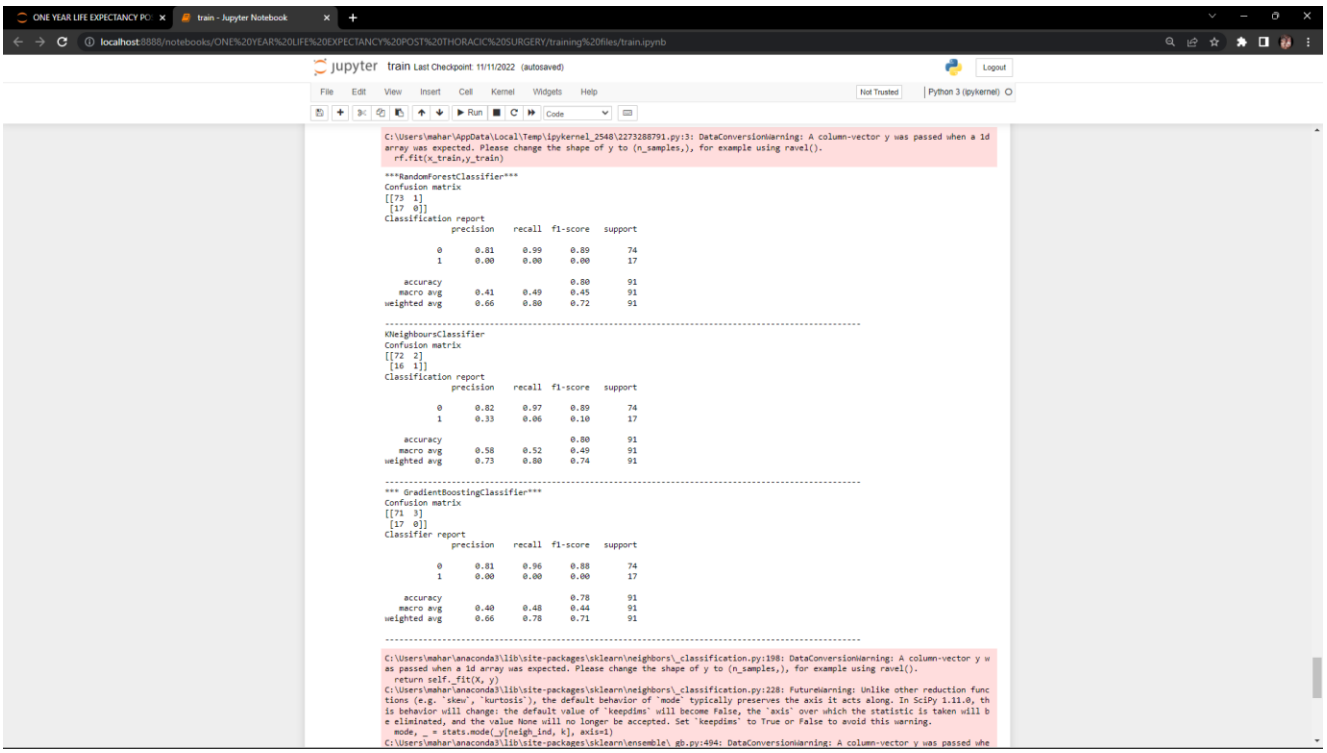


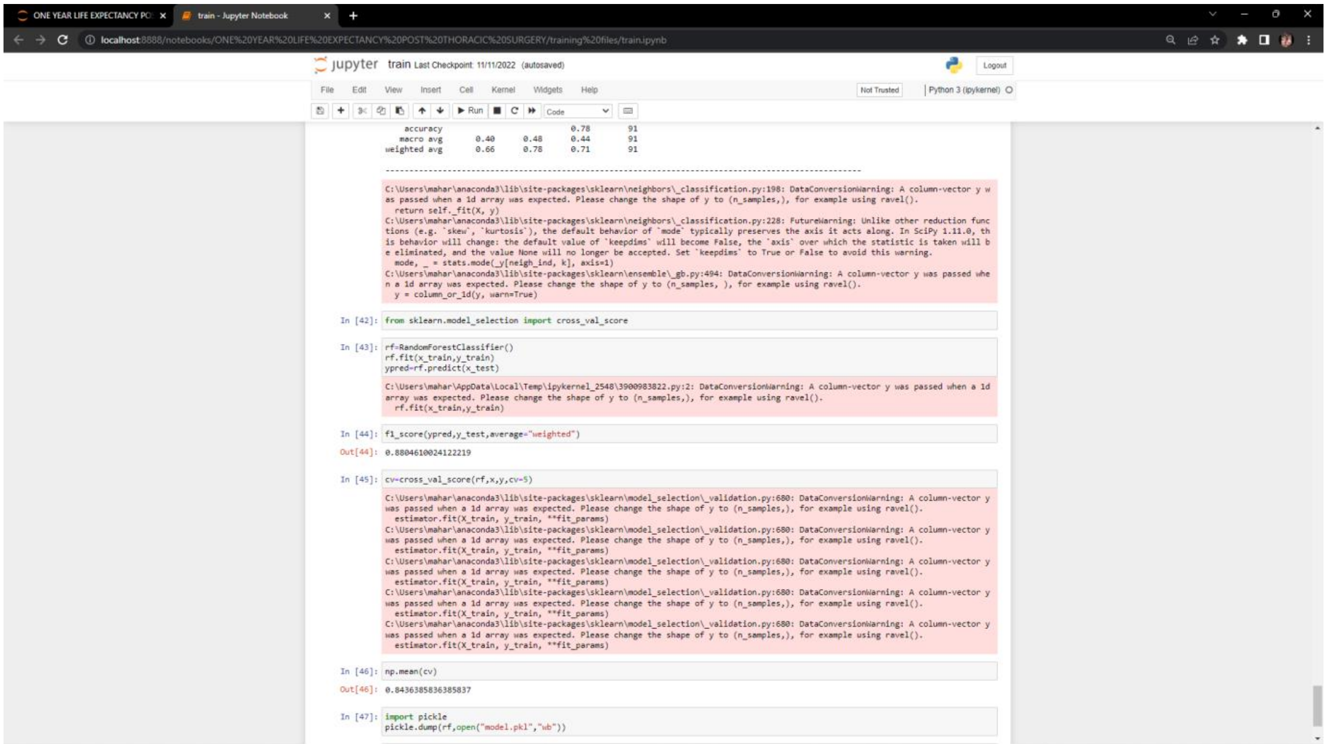
```
In [26]: df.isnull().sum()
Out[26]: Diagnosis           0
         FVC                 0
         FEV1                0
         Performance         0
         Pain                0
         Haemoptysis         0
         Dyspnoea            0
         Cough               0
         Weakness            0
         Tumor_Size          0
         Diabetes_Mellitus   0
         MI_6mo              0
         PAD                 0
         Smoking             0
         Asthma              0
         Age                 0
         Death_1yr           0
         dtype: int64

In [27]: df.drop(['FVC'],axis=1,inplace=True)

In [28]: x=df.iloc[:,0:15].values
         y=df.iloc[:,15:16].values

In [29]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)

In [30]: print("Shape of x_train {}".format(x_train.shape))
         print("Shape of y_train {}".format(y_train.shape))
         print("Shape of x_test {}".format(x_test.shape))
         print("Shape of y_test {}".format(y_test.shape))

         Shape of x_train (363, 15)
         Shape of y_train (363, 1)
         Shape of x_test (91, 15)
         Shape of y_test (91, 1)

In [31]: sc=StandardScaler()
         x_train=sc.fit_transform(x_train)
         x_test=sc.fit_transform(x_test)

In [32]: def decisionTree(x_train, x_test, y_train, y_test):
             dt= DecisionTreeClassifier()
             dt.fit(x_train,y_train)
             yPred = dt.predict(x_test)
             print("***DecisionTreeClassifier***")
             print('Confusion matrix')
             print (confusion_matrix(y_test,yPred))
             print('Classification report')
             print(classification_report(y_test, yPred))

In [33]: decisionTree(x_train, x_test, y_train, y_test)

         ***DecisionTreeClassifier***
         Confusion matrix
```


ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [33]: decisionTree(x_train, x_test, y_train, y_test)
DecisionTreeClassifier
Confusion matrix
[[62 12]
[12 5]]
Classification report
precision recall f1-score support
0 0.84 0.84 0.84 74
1 0.29 0.29 0.29 17
accuracy 0.74 91
macro avg 0.57 0.57 0.57 91
weighted avg 0.74 0.74 0.74 91
In [34]: def randomForest(x_train, x_test, y_train, y_test):
rf = RandomForestClassifier()
rf.fit(x_train,y_train)
yPred = rf.predict(x_test)
print('***RandomForestClassifier***')
print('Confusion matrix')
print(confusion_matrix(y_test,yPred))
print('Classification report')
print(classification_report(y_test,yPred))
In [35]: randomForest(x_train, x_test, y_train, y_test)
RandomForestClassifier
Confusion matrix
[[73 1]
[17 0]]
Classification report
precision recall f1-score support
0 0.81 0.99 0.89 74
1 0.00 0.00 0.00 17
accuracy 0.80 91
macro avg 0.41 0.49 0.45 91
weighted avg 0.66 0.80 0.72 91
C:\Users\mahar\AppData\Local\Temp\ipykernel_2548\2273288791.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
rf.fit(x_train,y_train)
In [36]: def KNN(x_train, x_test, y_train, y_test):
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)
yPred = knn.predict(x_test)
print('KNeighboursClassifier')
print('Confusion matrix')
print(confusion_matrix(y_test,yPred))
print('Classification report')
print(classification_report(y_test,yPred))


ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [36]: def KNN(x_train, x_test, y_train, y_test):
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)
yPred = knn.predict(x_test)
print('KNeighboursClassifier')
print('Confusion matrix')
print(confusion_matrix(y_test,yPred))
print('Classification report')
print(classification_report(y_test,yPred))
In [37]: KNN(x_train, x_test, y_train, y_test)
KNeighboursClassifier
Confusion matrix
[[72 2]
[16 1]]
Classification report
precision recall f1-score support
0 0.82 0.97 0.89 74
1 0.33 0.06 0.10 17
accuracy 0.80 91
macro avg 0.58 0.52 0.49 91
weighted avg 0.73 0.80 0.74 91
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\neighbors_classification.py:198: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
return self._fit(X, y)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\neighbors_classification.py:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
In [38]: def xgboost(x_train, x_test, y_train, y_test):
xg = GradientBoostingClassifier()
xg.fit(x_train,y_train)
yPred = xg.predict(x_test)
print('*** GradientBoostingClassifier***')
print('Confusion matrix')
print(confusion_matrix(y_test,yPred))
print('Classifier report')
print(classification_report(y_test,yPred))
In [39]: xgboost(x_train, x_test, y_train, y_test)
*** GradientBoostingClassifier***
Confusion matrix
[[70 4]
[17 0]]
Classifier report
precision recall f1-score support
0 0.80 0.95 0.87 74
1 0.00 0.00 0.00 17


ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [39]: xgboost(x_train, x_test, y_train, y_test)
*** GradientBoostingClassifier***
Confusion matrix
[[70 4]
[17 0]]
Classifier report
precision recall f1-score support
0 0.80 0.95 0.87 74
1 0.00 0.00 0.00 17
accuracy 0.77 91
macro avg 0.40 0.47 0.43 91
weighted avg 0.65 0.77 0.71 91
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\ensemble_gb.py:494: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
y = column_or_1d(y, warn=True)
In [40]: def compareModel(x_train, x_test, y_train, y_test):
decisionTree(x_train, x_test, y_train, y_test)
print('-'*100)
randomForest(x_train, x_test, y_train, y_test)
print('-'*100)
KNN(x_train, x_test, y_train, y_test)
print('-'*100)
xgboost(x_train, x_test, y_train, y_test)
print('-'*100)
In [41]: compareModel(x_train, x_test, y_train, y_test)
DecisionTreeClassifier
Confusion matrix
[[60 14]
[12 5]]
Classification report
precision recall f1-score support
0 0.83 0.81 0.82 74
1 0.26 0.29 0.28 17
accuracy 0.71 91
macro avg 0.55 0.55 0.55 91
weighted avg 0.73 0.71 0.72 91
C:\Users\mahar\AppData\Local\Temp\ipykernel_2548\2273288791.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
rf.fit(x_train,y_train)
RandomForestClassifier
Confusion matrix
[[73 1]
[17 0]]
Classification report
precision recall f1-score support


ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
C:\Users\mahar\AppData\Local\Temp\ipykernel_2548\2273288791.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
rf.fit(x_train,y_train)
RandomForestClassifier
Confusion matrix
[[73 1]
[17 0]]
Classification report
precision recall f1-score support
0 0.81 0.99 0.89 74
1 0.00 0.00 0.00 17
accuracy 0.80 91
macro avg 0.41 0.49 0.45 91
weighted avg 0.66 0.80 0.72 91
KNeighboursClassifier
Confusion matrix
[[72 2]
[16 1]]
Classification report
precision recall f1-score support
0 0.82 0.97 0.89 74
1 0.33 0.06 0.10 17
accuracy 0.80 91
macro avg 0.58 0.52 0.49 91
weighted avg 0.73 0.80 0.74 91
*** GradientBoostingClassifier***
Confusion matrix
[[71 3]
[17 0]]
Classifier report
precision recall f1-score support
0 0.81 0.96 0.88 74
1 0.00 0.00 0.00 17
accuracy 0.78 91
macro avg 0.40 0.48 0.44 91
weighted avg 0.66 0.78 0.71 91
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\neighbors_classification.py:198: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
return self._fit(X, y)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\neighbors_classification.py:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\ensemble_gb.py:494: DataConversionWarning: A column-vector y was passed whe



ONE YEAR LIFE EXPECTANCY PO
train - Jupyter Notebook
localhost:8888/notebooks/ONE%20YEAR%20LIFE%20EXPECTANCY%20POST%20THORACIC%20SURGERY/training%20files/train.ipynb
jupyter train Last Checkpoint: 11/11/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
accuracy 0.78 91
macro avg 0.40 0.48 0.44 91
weighted avg 0.66 0.78 0.71 91
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\neighbors_classification.py:198: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
return self._fit(X, y)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\neighbors_classification.py:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\ensemble_gb.py:494: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
y = column_or_1d(y, warn=True)
In [42]: from sklearn.model_selection import cross_val_score
In [43]: rf=RandomForestClassifier()
rf.fit(x_train,y_train)
ypred=rf.predict(x_test)
C:\Users\mahar\AppData\Local\Temp\ipykernel_2548\3900983822.py:2: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
rf.fit(x_train,y_train)
In [44]: f1_score(ypred,y_test,average="weighted")
Out[44]: 0.8804610024122219
In [45]: cv=cross_val_score(rf,x,y,cv=5)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\model_selection_validation.py:680: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
estimator.fit(X_train, y_train, **fit_params)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\model_selection_validation.py:680: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
estimator.fit(X_train, y_train, **fit_params)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\model_selection_validation.py:680: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
estimator.fit(X_train, y_train, **fit_params)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\model_selection_validation.py:680: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
estimator.fit(X_train, y_train, **fit_params)
C:\Users\mahar\anaconda3\lib\site-packages\sklearn\model_selection_validation.py:680: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
estimator.fit(X_train, y_train, **fit_params)
In [46]: np.mean(cv)
Out[46]: 0.8436385836385837
In [47]: import pickle
pickle.dump(rf,open("model.pkl","wb"))